REMARKS

Claims 1 – 12, 14-24 are pending in the application, and claim 13 is cancelled.

Claim Rejections – 35 USC § 102

In this section of the official action, Claims 1 – 24 are rejected under 35 USC 102(e), as being anticipated by Hines US Patent Application No. 2003/0028858 A1.

Favorable reconsideration of these rejections is respectfully requested in light of the above amendments and the following explanations.

As stated in the field of invention section of this application: "The present invention is of a system and method for visualization and debugging of *constraint systems,* and in particular, of a system and method for constraint resolution visualization that helps the user in case the results of the resolution process do not match the expected results".
As described in the background of the invention section of this application, in page 1, line 13: "*Constraint systems* are difficult to debug because they are *declarative* in nature, which means there is *no execution process as for imperative code.* Instead, constraints are evaluated by an algorithm which is obscure to the user".
The present invention teaches a visual debugging system and method for a constraint system which is encapsulated in a declarative language which is, by definition, devoid of an execution process.
With the present invention, the user is provided with a visual interface which is appropriately designed so as to facilitate visual tracing of the sequence of generation events where generation entities that are fields are assigned a specific value, changed with respect to their permissible values using an accordingly modified field associated constraint, assigned an order relation with another field(s) etc.

Hines relates to creation and use of graphical displays in connection with debugging concurrent or distributed software systems, as described in [0005]: "Evolution diagrams can be used to graphically display operation of a distributed or concurrent software system to a programmer to aid in debugging".

Claim 1 defines a debugger for visual debugging of a declarative language encapsulated constraint system, comprising: a collector for collecting *generation events* during a test process comprising *constraint resolution, generation objects, the generation objects comprising fields*, and the test process further comprising generation decisions, and a systematic graphical representation for relating the generation objects and the generation decisions, during the test process, *for debugging the declarative language encapsulated constraint system*.

The present invention teaches the novel and inventive idea of a debugger for the debugging of a *constraint system* which facilitates graphical representation for relating generation objects, comprising fields with the generation decisions, during the test process. Thus the debugger graphically presents the fields and the sequence of generation events where the fields may change with respect to their associated values, permissible values, order relations to other fields, etc. An example is provide in Fig. 2 where fields (gc#1, y, x) are presented on one axis, generation events (initial range, reduction, etc.) are presented on another axis, and the changes in fields upon the different events, such as in a permissible value range, are presented in an appropriate position with respect to the two axes, so as to trace the changes of a specific field upon a specific generation event.

Hines rather discloses graphical displays in connection with debugging concurrent or distributed software systems. As illustrated in Fig. 46, different components or

modules of the distributed software system are presented on one axis, and the sequence of events is accordingly positioned so as to show their order and the module or component where the events occur. However Hines, in the various graphical representations provided by him, never discloses or even hints at the idea of a graphical representation for relating the generation objects, comprising *fields*, and the generation decisions, during the test process *comprising resolution* of a constrained system which is encapsulated in a declarative language, as taught by the present invention.

It is thus respectfully believed that claim 1, as currently presented, is both novel and inventive over the prior art and should be allowed.

Claim 8 defines a method for visual debugging of *a constraint system*, the constraint system being encapsulated in a declarative language, comprising: displaying a plurality of generation events collected during a *constraint resolution* of the constraint system by a generator, such that a relationship between the plurality of generation events and *a plurality of generation entities comprising fields*, for generating the generation events is graphically displayed, and wherein an order of execution of the generation entities is also graphically displayed, for visual debugging of the group of constraints.

As described hereinabove, the present invention teaches the novel and inventive idea of a method for visual debugging of a *constraint system* which facilitates graphical representation for relating generation entities, comprising fields with the generation events, during the test process of the constraint system. Thus the method graphically presents *the fields* and the sequence of generation events where *the fields* may change

with respect to their associated values, permissible values, order relations to other fields, etc.

Hines rather discloses graphical displays in connection with debugging concurrent or distributed software systems. As illustrated in Fig. 46, different components or modules of the distributed software system are presented on one axis, and the sequence of events is accordingly positioned so as to show their order and the module or component where the events occur. However Hines never discloses or even hints at the idea of a graphical representation for relating the generation entities, comprising *fields*, and the generation events, during the test process *comprising resolution* of a constrained system which is encapsulated in a declarative language, as taught by the present invention.

It is thus respectfully believed that claim 8, as currently presented, is both novel and inventive over the prior art and should be allowed.

Claim 12 defines a method for debugging a generation process with a user, comprising: analyzing a generation process *comprising constraint resolution* of *a constraint system*, the constraint system being encapsulated in a declarative language, to extract a sequence of events from the generation process, and displaying at least a portion of the sequence of events to the user in *a visual display, wherein the visual display includes a representation of at least one generated field from at least one event.*

As described hereinabove, the present invention teaches the novel and inventive idea of a method for debugging a generation process with a user, comprising: analyzing a generation process of a constraint system, which facilitates *graphical representation for relating fields with the generation events, during the test process of the constraint*

system. Thus the method graphically presents *the fields* and the sequence of generation events where the fields may change with respect to their associated values, permissible values, order relations to other fields, etc.

Hines rather discloses graphical displays in connection with debugging concurrent or distributed software systems. As illustrated in Fig. 46, different components or modules of the distributed software system are presented on one axis, and the sequence of events is accordingly positioned so as to show their order and the module or component where the events occur. However Hines never discloses or even hints at the idea of a graphical representation for relating *fields*, and the generation events, during the test process *comprising resolution-* of a constrained system which is encapsulated in a declarative language, as taught by the present invention.

It is thus respectfully believed that claim 12, as currently presented, is both novel and inventive over the prior art and should be allowed.

Claim 17 defines a debugger for debugging a generation process, comprising: an analyzer for analyzing a generation process *comprising resolution of a constraint system*, the constraint system being encapsulated in a declarative language, to extract a sequence of events from the generation process, and *a visual display for displaying information related to at least one field*, and one of a constraint, a generation event, a path of a generation event, and a combination thereof.

The present invention teaches the novel and inventive idea of a debugger for debugging a generation process of a constraint system which is encapsulated in a declarative language, where an analyzer is used for visually displaying information *related to at least one field* as explained hereinabove.

Hines rather discloses graphical displays in connection with debugging concurrent or distributed software systems. As illustrated in Fig. 46, different components or modules of the distributed software system are presented on one axis, and the sequence of events is accordingly positioned so as to show their order and the module or component where the events occur. However Hines never discloses or even hints at the idea of a graphical representation for relating *fields*, and the generation events, during the test process *comprising resolution* of a constrained system which is encapsulated in a declarative language, as taught by the present invention.

It is thus respectfully believed that claim 17, as currently presented, is both novel and inventive over the prior art and should be allowed.

The remaining claims are believed to be allowable as being dependent on allowable main claims.

All of the matters raised by the Examiner have been dealt with and are believed to have been overcome. In view of the foregoing, it is respectfully submitted that all the claims now pending in the application are allowable over the cited reference.

An early Notice of Allowance is therefore respectfully requested.

Respectfully submitted,

Martin O. Moynihan

Martin Moynihan
Registration No. 40,338

Date: August 31, 2005